

Cricket Ball Prediction Model V2: Unified Heterogeneous Graph Architecture

Architecture Documentation

December 16, 2025

Abstract

This document describes Version 2 of the cricket ball prediction model. The key innovation is representing **all information as a single heterogeneous graph**, eliminating the separation between spatial and temporal processing. This enables full innings history while maintaining computational efficiency.

Contents

1	Overview	2
1.1	The Core Idea	2
1.2	Why This Matters	2
2	The Unified Graph Structure	3
2.1	Node Types	3
2.2	Edge Types	3
3	Full Graph Visualization	4
4	Why Full History Now Works	5
4.1	The Quadratic Problem (V1)	5
4.2	The Linear Solution (V2)	5
4.3	Efficiency Comparison	5
5	Model Architecture	6
5.1	Three-Stage Pipeline	6
5.2	Convolution Choices per Edge Type	6
6	Temporal Edge Structure	7
6.1	Three Types of Ball-to-Ball Edges	7
6.2	Why This Structure Matters	7
7	The Actor Matchup Graph	8
8	Data Flow Summary	8
9	Implementation Summary	9
9.1	Key PyTorch Geometric Components	9
9.2	Model Configuration	9
9.3	Training Details	9
10	Interpretability Benefits	9

1 Overview

1.1 The Core Idea

Instead of separate spatial and temporal streams, **everything becomes one heterogeneous graph**.

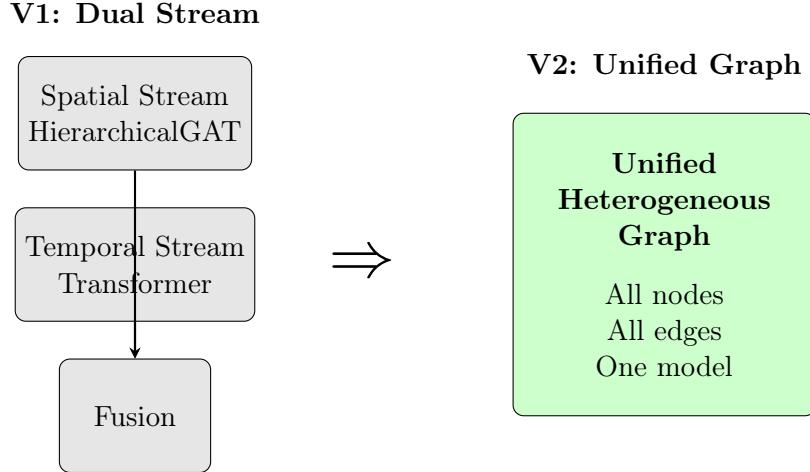


Figure 1: Architecture evolution from V1 (dual stream) to V2 (unified graph)

1.2 Why This Matters

Aspect	V1 (Dual Stream)	V2 (Unified Graph)
History length	Fixed 24 balls	Full innings
Complexity	$O(n^2)$ attention	$O(\text{edges})$
Same-bowler patterns	Soft attention bias	Explicit edges
Information flow	Separate then fuse	Unified message passing
Framework	Mixed PyTorch/PyG	Full PyTorch Geometric

2 The Unified Graph Structure

2.1 Node Types

The graph contains **6 node type categories**:

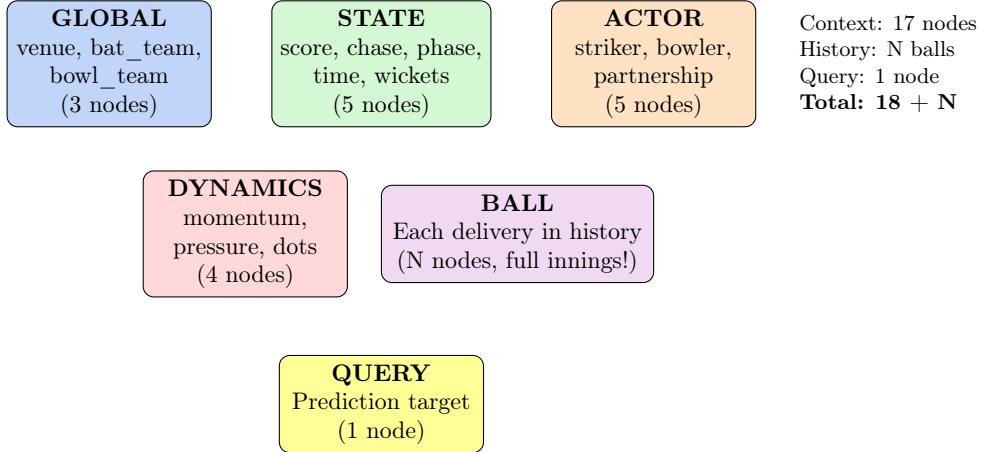


Figure 2: Six node type categories in the unified graph

2.2 Edge Types

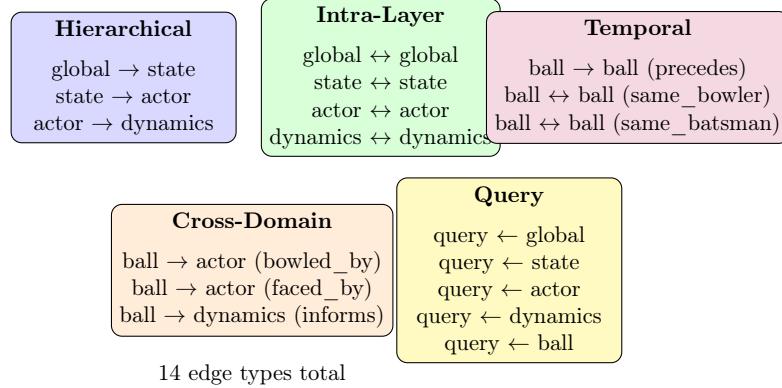


Figure 3: Edge type categories encoding different relationships

3 Full Graph Visualization

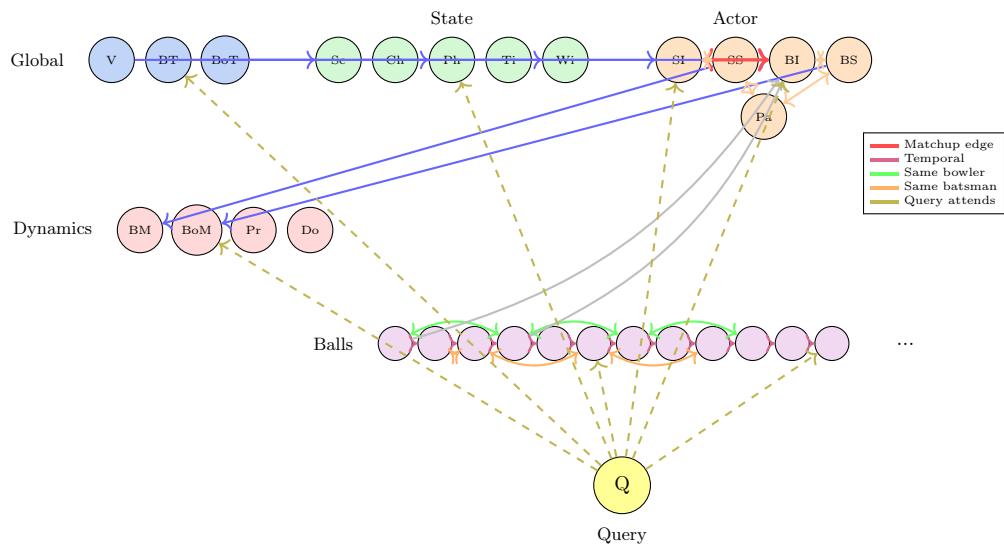
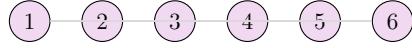


Figure 4: Complete unified graph structure showing all node and edge types

4 Why Full History Now Works

4.1 The Quadratic Problem (V1)

In V1's Transformer, every ball attends to every other ball:



6 balls \rightarrow 15 attention pairs

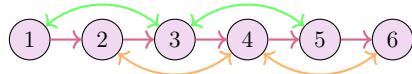
24 balls \rightarrow 276 pairs

120 balls \rightarrow 7,140 pairs

Figure 5: V1: Full attention scales $O(n^2)$

4.2 The Linear Solution (V2)

In V2's graph, attention only flows along edges:



6 balls \rightarrow 11 edges

24 balls \rightarrow \sim 100 edges

120 balls \rightarrow \sim 500 edges

Figure 6: V2: Sparse edges scale $O(n)$

4.3 Efficiency Comparison

History Length	V1 Attention Pairs	V2 Edges	Speedup
24 balls	576	\sim 150	4x
60 balls	3,600	\sim 350	10x
120 balls (full innings)	14,400	\sim 700	20x

Table 1: Computational comparison: V1 vs V2

5 Model Architecture

5.1 Three-Stage Pipeline

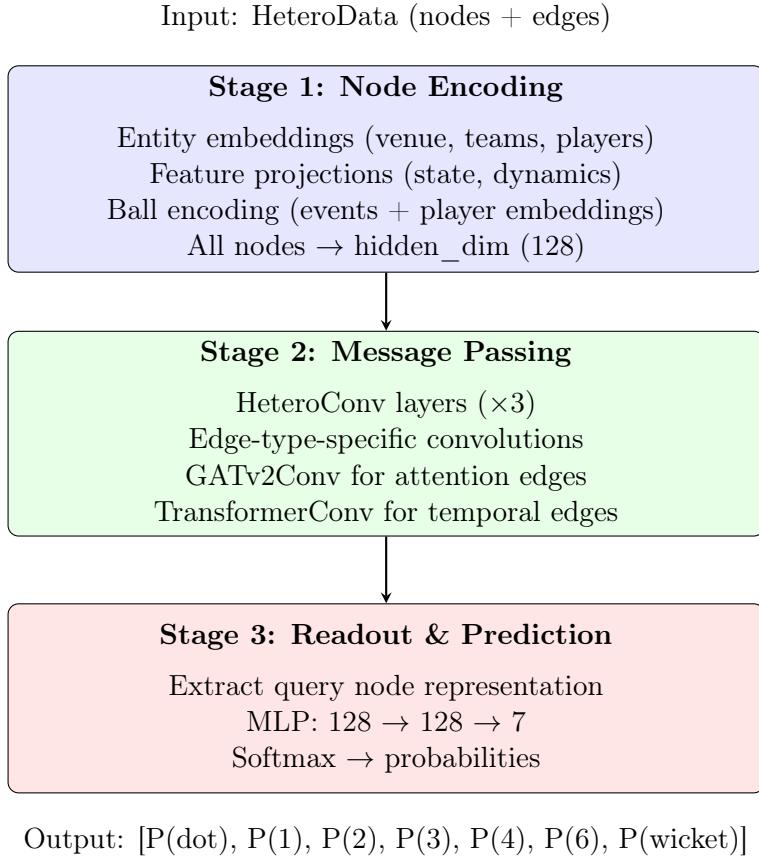


Figure 7: Three-stage model pipeline

5.2 Convolution Choices per Edge Type

Edge Type	Convolution	Rationale
Hierarchical	GATv2Conv	Attention learns which context matters
Intra-layer	GATv2Conv	Self-attention for permutation equivariance
Actor matchup	GATv2Conv	Learn matchup dynamics
Temporal (precedes)	TransformerConv	Edge features for temporal distance
Same-bowler/batsman	GATv2Conv	Aggregate spell patterns
Cross-domain	SAGEConv	Simple aggregation
Query	GATv2Conv	Learn what to attend for prediction

Table 2: Edge-type-specific convolution operators

6 Temporal Edge Structure

6.1 Three Types of Ball-to-Ball Edges

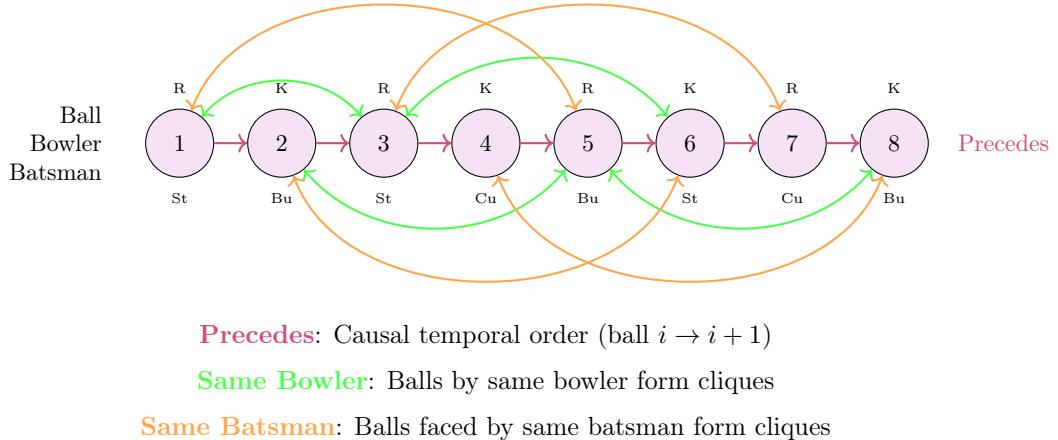


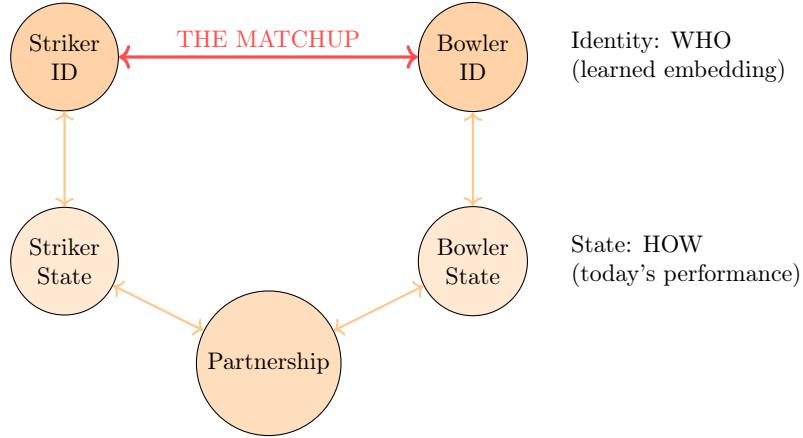
Figure 8: Temporal edge structure enables efficient pattern learning

6.2 Why This Structure Matters

- **Precedes:** Captures immediate context, momentum shifts
- **Same Bowler:** How is this bowler's spell going? Patterns in their deliveries
- **Same Batsman:** How is this batsman building their innings?

In V1, these patterns were captured via soft attention biases.
In V2, they are **explicit graph structure** that the model must respect.

7 The Actor Matchup Graph



The **Striker \leftrightarrow Bowler** edge is crucial:
 “Rohit vs Bumrah” encodes their historical confrontation

Figure 9: Actor layer graph structure (same as V1, now explicit)

8 Data Flow Summary

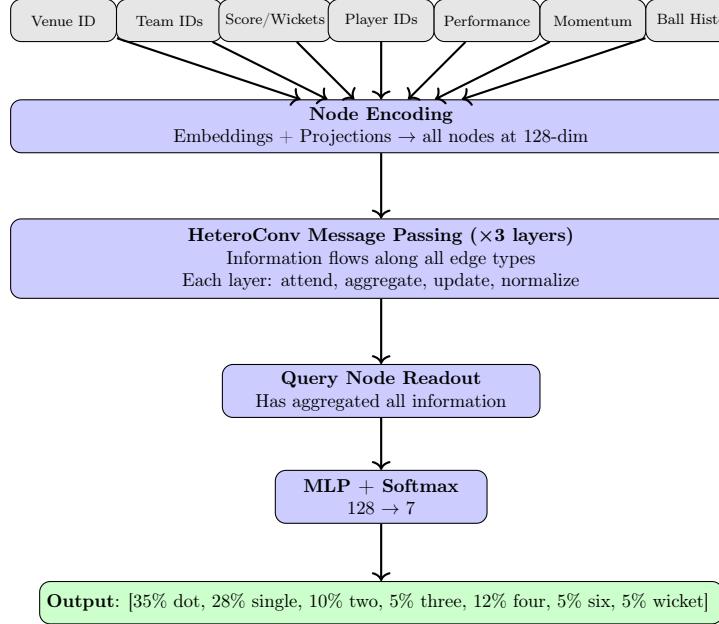


Figure 10: Complete data flow from raw inputs to prediction

9 Implementation Summary

9.1 Key PyTorch Geometric Components

Component	PyG Class
Data structure	HeteroData
Message passing	HeteroConv
Attention convolution	GATv2Conv
Temporal convolution	TransformerConv
Simple aggregation	SAGEConv
Batching	DataLoader (auto-batches HeteroData)

9.2 Model Configuration

Parameter	Value
Hidden dimension	128
Number of layers	3
Attention heads	4
Dropout	0.1
Number of classes	7
Estimated parameters	~720K

9.3 Training Details

- **Loss:** Cross-entropy with class weights (imbalanced outcomes)
- **Optimizer:** AdamW (lr=1e-3, weight_decay=0.01)
- **Scheduler:** Cosine annealing
- **Early stopping:** Patience of 10 epochs

10 Interpretability Benefits

The unified graph provides natural interpretability:

1. **Edge attention weights:** Which historical balls mattered most?
2. **Same-bowler attention:** How much did the bowler's spell inform the prediction?
3. **Matchup attention:** How important was the striker-bowler confrontation?
4. **Hierarchical attention:** Did global context (venue) or local dynamics drive the prediction?

All of these are directly extractable from the GATv2Conv attention weights.